DART

2020

VSCODE

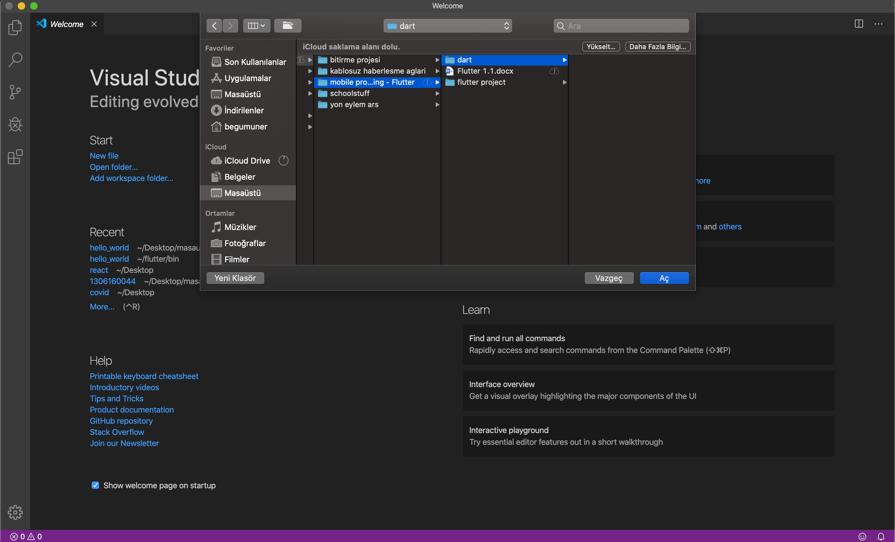
BEGUM UNER

Istanbul University

****

**DART- VsCode**

* Flutter Setup dokümanında kurulumları anlatmıştım, onları yaptıktan sonra şu an ilk dart dersi ile başlıyoruz. Bu derslerde Vscode kullanacağız. İlk olarak boş bir dart dosyası oluşturup dart dilini öğreneceğiz daha sonra Flutter’a geçeceğiz.
* 1.Adım:
* Bilgisayarımızda istediğimiz yere bir yeni klasör oluşturuyoruz. Ben ismini dart koydum.
* Daha sonra Vscode’u açıp Open folder diyoruz ve o oluşturduğumuz dosyayı açıyoruz.

****

* Yanda açılan Explorer kısmında kendi dosyamızın altında .dart uzantılı bir dosya oluşturuyoruz. İsmi size kalmış, ben test.dart koydum.

**ekran görüntüsü, ekran, oturma, telefon içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* Test.dart dosyamızın içine girip start debugging yapıyoruz.

**ekran görüntüsü, ekran, bilgisayar, dizüstü içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* Start debugging yaptığımızda bizim launch.json dosyamız oluşuyor.

**ekran görüntüsü, ekran, dizüstü, oturma içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* Test.dart klasörümüzün içine gidip main yazıp tab’a basıyoruz ve bizim dart şablonumuz oluşuyor ben içine print ile merhaba yazdırdım.

**ekran görüntüsü, ekran, elektronik eşyalar, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* Bu kodu start debugging yaparsak hata alırız çünkü launch.json dosyamızda “program”: “bin/main.dart” yazıyor bu demek oluyorki bin dosyasının içinde main metodu arıyor biz orayı “program”: “test.dart” yapıyor.

**ekran görüntüsü, ekran, bilgisayar, oturma içeren bir resim

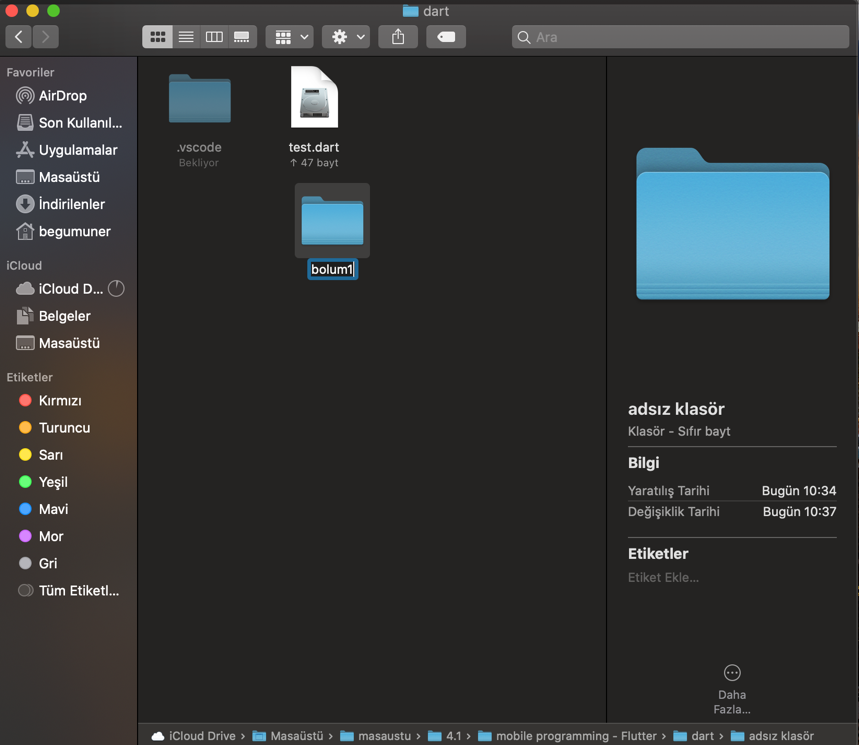
Açıklama otomatik olarak oluşturuldu**

* Bunu yapınca bizim test.dart dosyamızı start debugging yapıyoruz ve çalışıyor. Debug console’a merhaba yazdırıyor.

**ekran görüntüsü, ekran, elektronik eşyalar, bilgisayar içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* Daha sonra ilk oluşturduğumuz dart klasörümüzün içine gidip bir tane daha dosya oluşturuyorum ben adını bolum1 koydum

****

* Bu oluşturduğum bolum1 dosyasını Vscode’da görebiliyorum bu dosyayı isterseniz direk Vscode’da oluşturabilirsiniz. Daha sonra bu bolum1 dosyamın altında main.dart dosyası oluşturuyorum.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* Tekrar launch dosyama gidip program kısmıni değiştiriyorum çünkü kodumu bolum1’in altinda yazmak istiyorum.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

* Ve bolum1’in altındaki main.dart’ın içine main yazıp tab’a basıyorum ve içine print ile bolum1 yazdırıyorum start debugging yapıp çalıştırıyorum ve çalışıyor.

**ekran görüntüsü, ekran, bilgisayar, dizüstü içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**VERI TIPLERI VE DEĞİŞKENLER**

**TEMEL VERI TIPLERI**

* **Numbers** (Sayılar)
  + **Integer**– Tam Sayılar
  + **Double** – Ondalıklı Sayılar
* **Strings (**Metinler)
* **Booleans** (True/False)
* **Lists**
* **Maps**
* **Final**
* **Const**
* String, integer gibi genel veri tiplerinden yanı sıra olarak kullandığımız farklı; önemli 2 veri tipinden bahsedeceğiz.
* **Var** veri tipi, belli bir değere sahip değildir. İstediğimiz değişkeni atayabiliriz, fark etmez. Var ile tanımlanan değişkene dilerseniz string dilerseniz integer değer verebilirsiniz.
* **Dynamic** veri tipi, değişken bir yapıya sahiptir. Dynamic ile oluşturduğumuz değişkeni ilk string veri tipinde tanımlayıp daha sonra aynı değişkene integer bir değer atayabiliyoruz. Tabi bu string ve integer örneğimiz, en yaygın değişken tipleri olduğu için sürekli kullanıyorum. Double, bool gibi değişkenler de kullanılabilir.

**!** Bellekte ayrılmış boş bir alan Null’dır.

**!** Değişken isimlendirmelerinde ilk harf küçük sonraki kelimenin ilk harfi büyük olur

ÖRNEK :

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**List**

Dart programlama dilinde, *Array yapısı yoktur.* Bunun yerine **List** yapısını kullanırız. Birden fazla veriyi tutmak için kullandığımız bu **List** yapısı da ikiye ayrılır:

* Sabit uzunluklu
* Büyüyen yapıya sahip

Aşağıda bu iki özelliğiyle de tanımlama şekillerini görebilirsiniz. **List** yapısı, alacağı değerlere göre değişken veri tipi alabilir. Veriler, sıralıdır. **List** yapısına veri eklemek, silmek gibi işlevleri hazır methodları ile kolayca yapabiliriz.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**List Methodlar**

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Maps**

Map yapısı da birden fazla veriyi tutmamıza yarayan liste türlerinden birisidir. Map’in en önemli özellikleri:

* **Key – value şeklinde çalışır.**
* **Sırasızdır.**

Bu yapıyı da sözlük olarak düşünebiliriz. Bir kelimenin sadece bir karşılığı vardır. Burda da ***kelime=key***, kelimeye karşılık gelen ***anlam=value*** oluyor.

Aynı zamanda Map yapısını *veritabanı kullanırken* de çok kullandığımız için önemlidir. Aşağıda map yapısının kullanımlarına örnekler bulunmaktadır.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Final**

Oluşturulan değişken sabittir, daha sonra farklı bir değer atanamaz. Değer **kullanıldığı** andan itibaren ram üzerinde tutulmaya başlar. Değer tipini belirtmek gerekir.

**Const**

Oluşturulan değer final'da olduğu gibi sabittir. Ancak **oluşturulduğu** andan itibaren ram üzerinde tutulmaya başlanır. Tip belirtmeye gerek duymaz, değer tipine göre tip ataması yapılır.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**If – else yapısı**

Kısaca hangi if() içindeki şart true ise sadece o blok çalışır ve if ten çıkılır, true  değilse yani false ise diğer bloklar kontrol edilir.

if(şart) içerisindeki şart doğru ise yani şart sağlanıyorsa bu true anlamına gelir ve if altındaki ilk blok çalışır diğer if else şartlarına bakılmaz. Alttaki if else şartları true verse bile ilk true veren if çalışır.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Ternary if kullanimi**

Uzun uzun if yazmaktansa ternary kullanarak kısaltabiliyoruz.

**ekran görüntüsü, ekran, televizyon, oturma içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**saat, metre, cadde, işaret içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Switch- case yapısı**

Switch içerisindeki değişkenin değeri hangi case ifadesinde ise o blok çalışır ve break ile dışarı fırlar yani gerisi kontrol edilmez.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**For Döngüsü**

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

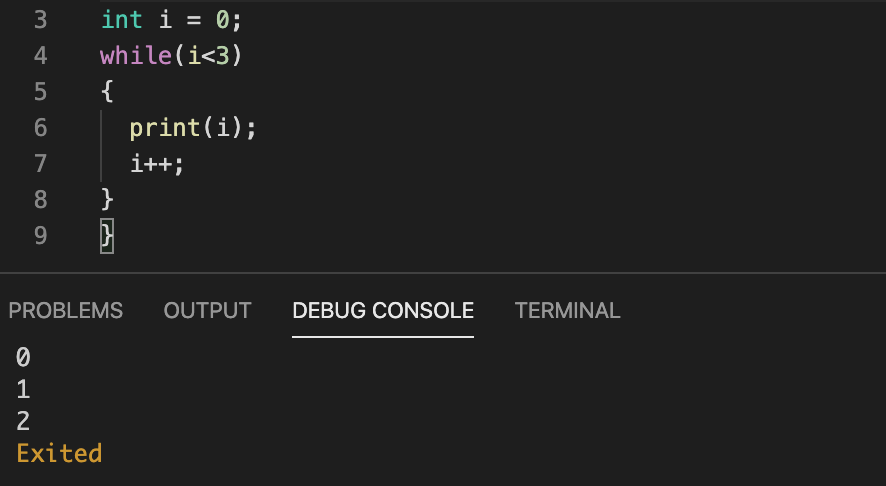
**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

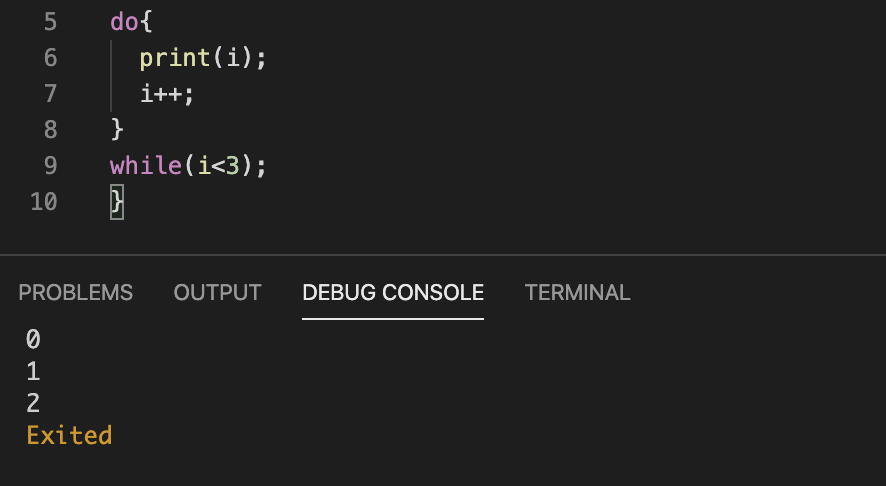
metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**While Döngüsü**



**Do-While Döngüsü**

****

**Fonksiyonlar**

Fonksiyon, matematikte değişken sayıları girdi olarak kabul edip bunlardan bir çıktı sayısı oluşmasını sağlayan kurallardır.

! isimlendirmede ilk harf küçük sonraki kelimenin ilk harfi büyük olur

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Optional ve named parametreler**

**!** Method overloading dart’ta yok.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Hata yakalama**

**Try**: Hatayı yakalar. Bir hataya neden olabilecek kodlar bu blok içine yazılır. Burada bir hata/istisna meydana geldiğinde catch bloğu çalışır.

**On**: Hata tipi belirtilmesi gerektiğinde **on** bloğu kullanılır.

**Catch:** hata olursa bu blok içindeki kodlar çalışır.

**Finally**: Hata olsa da olmasa da çalıştırılacak kod bloğudur.

**Hata/İstisna Türleri**

**DeferredLoadException: Ertelenmiş** bir kütüphane yüklenemediğinde atılan istisna.

**FormatException:**Bir string veya başka bir veri beklenen bir formata sahip olmadığında ve ayrıştırılamadığında veya işlenemediğinde ortaya çıkan istisna.

**IntegerDivisionByZeroException:**Bir sayı sıfıra bölündüğünde atılır.

**IOException:**Tüm Girdi-Çıktıyla ilgili istisnalar için temel sınıf.

**IsolateSpawnException:** Bir izolat oluşturulamadığında atılır.

**Timeout:**Asenkron (Zaman uyumsuz) bir sonuç beklenirken zaman aşımı gerçekleştiğinde atılır.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Nesneye yönelik programlama**

Nesne yönelimli programlama dillerinde, tüm sınıflar birer nesne ile ifade edilmektedir. Dart programlama dili de nesneye yönelik bir programlama dili olduğu için bu konu Dart dilinin temelini oluşturmaktadır. Nesne yönelimliden kasıt şöyledir. Örneğin; bir okul uygulaması yazacaksınız. Burada okul, sınıflar, sıralar, öğrenciler vb. her şey birer nesnedir. Ve programlama tarafında bunların hepsi ayrı ayrı sınıflar tarafından temsil edilir.

**Class**

**Class syntax:**

class class\_name {  
// Properties (Variables  
// Constructor   
// Functions   
// Getters and Setters   
}

**!** isimlendirmede ilk harf büyük

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Constructor**

Bir sınıftan nesne oluşturduğumuz anda çalışmasını istediğimiz sınıf fonksiyonuna yapici fonksiyon (constructor) denir. Her sınıfın vardır.  Constructor’lar değer döndüremezler fakat gövdelerine işler yazılabilir.  Constructor’lar tanımlansın ya da tanımlanmasın değer tiplerine sıfır, referans tiplerine "null" değerini atayan varsayılan bir yapılandırıcı vardır. Yapılandırıcısı tanımlandıktan sonra varsayılan yapılandırıcı bir daha kullanılmaz. Yapılandırıcıların bilinen temel özellikleri:

* Kendi sınıfı ile aynı isme sahip olması,
* Genellikle açık bir dönüş tipi olmaması,
* Başka sınıflar tarafından kullanılabilmesi için erişimin public (aleni) olmasıdır.
* Bir sınıfta 1 tane cons olur.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

# **Getters ve Setters Metodları**

* Getter, sınıf alanının verilerini okumak veya almak için kullanılırken
* Setter, sınıf alanının verilerini bir değişkene ayarlamak için kullanılır.

**GET**

return\_type **get** field\_name{

...

}

**SET**

**set** field\_name{

...

}

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Private**

Değişkenin önüne “\_” koyarak onu private yapabiliriz.

****

**Kalıtım**

* Kalıtım günlük yaşamdan da bildiğimiz ortak özelliklerin alt yapılara aktarılabilmesi işlemidir.
* Yani şöyle bir örnek düşünelim. Bir kadın ve erkek var. Bunlar evlendiklerinde çocuklarına kendilerinde var olan özellikleri aktarırlar. Aynı zamanda çocuk kendine has özellikler geliştirir. İşte bu ortak özellikleri her defasında her yeni çocukta yazmak yerine anne ve babadan kalıtabiliriz. İşte bu yapının aynısını programlamada kullanabiliyoruz.
* Dart’ta unutulmaması gereken konu şudur. Bir sınıf sadece bir üst sınıftan türeyip özelliklerini alabilir. Birden fazla sınıftan türetme yapamayız.
* Ne zaman kalıtım kullanmalıyım ne zaman kullanmamalıyım. Birbiri ile bağıntılı olup, birbirinden türeyebilen sınıflarınız varsa mutlaka kalıtım kullanmalısınız.
* Bu yine hem daha az kod yazmanıza olanak sağlayacak, aynı zamanda tek kaynaktan daha kolay kod bakımı ve güncellemesine imkân verecektir.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Overriding**

* Nesne Yönelimli (Object Oriented) Programlama'da, ana sınıftaki bazı metodlara, o sınıftan türetilen yeni sınıflarda ihtiyaç duyulmayabilir ya da bu metodlar üzerinde değişiklikler yapılması gerekebilir. Bu durum genellikle yazılan kodun, bir temel sınıf ve bu sınıftan türetilmiş farklı sınıfları etkileyeceği zamanlarda ortaya çıkar ve **Overriding (geçersiz kılma)** kullanılarak çözülebilir
* Extend ettiğiniz sınıftaki aynı metodu içeri farklı şekilde kendi sınıfında kullanmak.
* ‘super()’ anahtar kelimesi 1 üst sınıfa git demektir.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Polimorfizm**

* Nesne yönelimli programlama dillerinde çok biçimlilik aynı temel sınıftan türetilmiş olan yeni sınıflarda metotların farklı şekillerde davranabilmesidir.
* Polimorfizm bir varlığın veya bir nesnenin birden fazla forma sahip olmasıdır denilebilir.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Kalıtımda Constructor kullanımı**

* İlk üst sınıfın constructor’ı çalışır. Sonra kendi constructor’ı çalışır.
* Parametresiz constructor’larda sıkıntı olmuyor gizli bir super keyword’u taşıyorlar.
* Parametre alınca alt sınıftaki super anahtar kelimesine de parametre vermeliyiz.

**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Abstract class**

* İçindeki methodların içi boştur (gövdesiz) sadece tanımlamaları vardır.
* Nesne üretilmez.
* Abstract Class, ortak özellikli olan Class’lara base class olma rolünü üstlenir. Mesela bir class’ınız var ve bu class da pek çok ortak yön olarak özellikleriniz var. Aynı özellikleride içinde tutan başka bir classs’ınız daha var. Tekrardan aynı özellerini yazmak yerine base (yani abstract) class tanımlayıp buradan türetilmiş sınıflara kodumuzun daha efektif olmasını sağlıyoruz.
* Aynı özellikleri başka sınıflara da base alarak oluşturmak istediğimiz için bu class’lara abstract class adını veriyoruz

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**İnterface**

* Aralarında kalıtımsal bir şey bulunmayan ama ortak özellikleri olan sınıflar için kullanılır
* Dart’ta interface yoktur. Biz class’ları o şekilde kullanıyoruz.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Static Keyword**

* Oluşturduğumuz sınıfta bazı durumlarda başka bir metotta, sınıfta ulaşmamız, değer atamamız gerekebilir. Bu gibi durumlar için statik değişken tanımlaması yaparız.
* Örneğin bir Okul sınıfı oluşturduğumuzda, okuldaki öğrenci sayısını içeren değişkeni ana fonksiyon içerisinde belirtebiliriz.

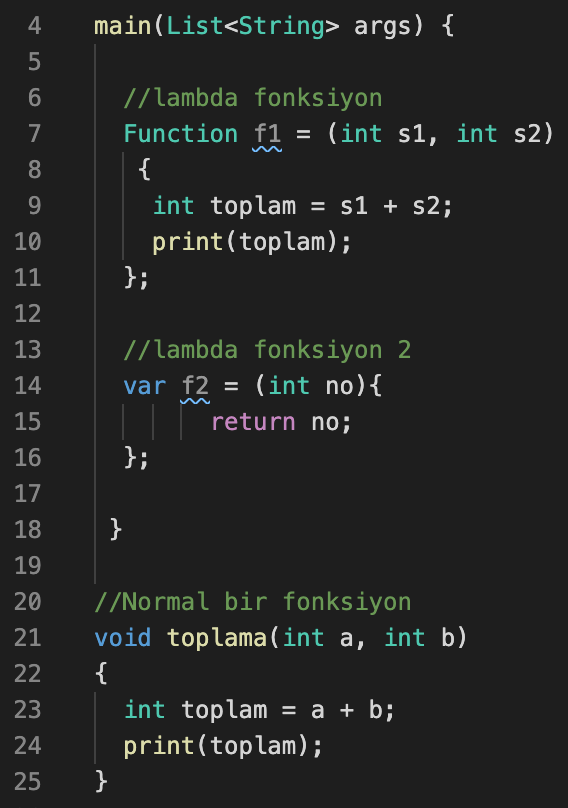
**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu**

**Fonksiyonel Programlama**

**Lambda fonksiyon:**

Lambda: ismi olmayan fonksiyonlardır. Dart’ta her fonksiyon bir nesnedir.



**Higher Order fonksiyon:**

Bir fonksiyonu parametre olarak alan veya geriye fonksiyon döndüren fonksiyonlardır.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu